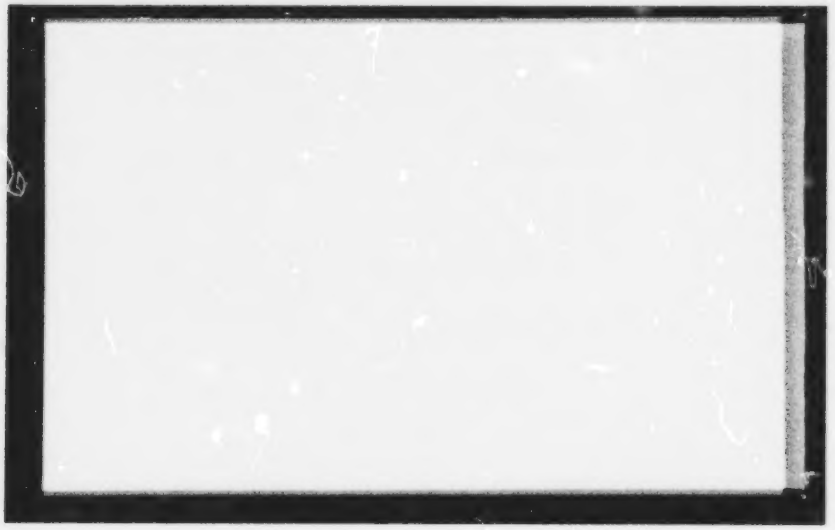
UNIVERSITÉ DE MONTRÉAL



CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, STATION A, MONTRÉAL, QUÉBEC, H3C 3J7

A GREEDY APPROACH FOR INTEGER PROGRAMMING

by

Jean-Pierre Dussault
and Jacques-A. Ferland

This research was supported by the National Research Council of Canada, Grant A8312. The first author had a scholarship from the *Ministère de l'Éducation du Québec* to complete this research.

Université de Montréal

Centre de recherche sur les transports - Publication # 148
Département d'informatique et de
recherche opérationnelle - Publication # 337

Septembre 1979

Dépôt légal - 4e trimestre 1979

ABSTRACT

A greedy approach is applied to develop heuristics solving some integer programs. A general procedure is established, and then, for three sample problems, we define a straightforward and efficient heuristic. A bound on the distance from optimality is available in each case. Numerical results and comparisons are reported.

1. Introduction

Let the Primal Integer Program be:

$$\begin{aligned} (P.I.P.) \qquad & \text{Min } f(x) \\ & \text{s.t. } f_i(x) \le b_i \qquad 1 \le i \le m \\ & x \in X \subset Z^n \quad , \quad |X| \text{ finite.} \end{aligned}$$

where $Z^n = \{x \in \mathbb{R}^n : x_j \text{ is integer}, 1 \le j \le n\}$. A Lagrangean dual of this problem is:

$$\begin{aligned} (D.I.P.) \qquad & \text{Max } D(\lambda) \\ & \lambda \ge 0 \end{aligned}$$

where

$$(1.1) \qquad D(\lambda) = \min_{x \in X} \{f(x) + \sum_{i=1}^{m} \lambda_i \, (f_i(x) - b_i)\}.$$

For a fixed value of λ, let $x(\lambda) \in X$ be a solution of (1.1). A subgradient of D at λ is given by the vector g specified by $g_i = (f_i(x(\lambda) - b_i)$, $1 \le i \le m$. This observation is at the origin of this procedure to solve the dual (D.I.P.):

Step 0: Determine an initial feasible solution λ for (D.I.P.).

Step 1: For the specified value of λ, determine $X(\lambda)$, solution of (1.1).

Step 2: If the stopping criterion is satisfied, then STOP.

Step 3: Determine

a) a direction of improvement for D using a subgradient;

b) a stepsize in this direction to move away from λ.

Go to Step 1 with this new dual-solution.

Notice that the three problems to be analysed belong to the class of 0-1

linear programs. Hence, once the coefficients of each primal variable are regrouped, Step 1 is trivial to perform.

A subgradient optimization technique can be used to determine a direction and stepsize in Step 3. The direction is a subgradient of D at λ. Polyak [7,8], Held *et al.* [5] and Goffin [4] studied several stepsizes to insure convergence and improve its rate. They prove convergence of the procedure, but the objective D is not monotone.

We prefer to go along Klastorin's approach [6] where the direction and the stepsize are selected to "try" maintaining the monotonicity of the objective, D. In order to track down the variations of the objective easily, only one dual variable is modified at each iteration. We use a greedy criterion to modify the dual variable having the largest marginal improvement rate. (In differentiable optimization, this direction is called Gauss-Southwell.) The stepsize is selected large enough to change the state of a primal variable from 0 to 1 or 1 to 0. Unfortunately this stepsize does not guarantee the monotonicity of the objective D.

Step 2 must be worked out for each problem. If a primal-feasible solution is available at each iteration, then the difference between the values of the objective functions of (P.I.P.) and (D.I.P.) can be used as a stopping criterion. This is a conservative criterion since these problems are not convex, and duality gap may exist.

Numerical experiments show the efficiency of the method.

2. Generalized Assignment Problem (G.A.P.)

This section summarizes the work of Klastorin [6] on the G.A.P.

The G.A.P. is specified as follows:

$$\min \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}\, x_{ij}$$

$$\text{(2.1)} \qquad \text{s.t.} \quad \sum_{j=1}^{n} r_{ij}\, x_{ij} \le b_i \,, \qquad 1 \le i \le m,$$

$$\sum_{i=1}^{m} x_{ij} = 1 \,, \qquad 1 \le j \le n,$$

$$x_{ij} = 0 \text{ or } 1 \,, \qquad 1 \le i \le m, \quad 1 \le j \le n.$$

As an illustration, consider the assignment of administrative offices (j) into buildings (i) where office j occupies a surface r_{ij} in building i, b_i is the surface available in building i and the assignment cost of office j into building i is c_{ij}.

A Lagrangean dual of (2.1) is:

$$\text{(2.2)} \qquad \max_{\lambda \ge 0} \quad D(\lambda)$$

where

$$\text{(2.3)} \qquad D(\lambda) = \min_{x \in X} \left\{ \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}\, x_{ij} + \sum_{i=1}^{m} \lambda_i \left(\sum_{j=1}^{n} r_{ij}\, x_{ij} - b_i \right) \right\}$$

and $$X = \{x: \sum_{i=1}^{m} x_{ij} = 1,\ 1 \le j \le n,\ x_{ij} = 0 \text{ or } 1,\ 1 \le j \le n,\ 1 \le i \le m\} \,.$$

For a fixed value of λ, Step 1 is easily performed if (2.3) is rewritten as

$$(2.4) \qquad D(\lambda) = \min_{x \in X} \{ \sum_{j=1}^{n} \sum_{i=1}^{m} (c_{ij} + \lambda_i r_{ij}) x_{ij} - \sum_{i=1}^{m} \lambda_i b_i \} .$$

Indeed, for each j, $1 \le j \le n$, determine an index i_j such as

$$c_{i_j j} + \lambda_{i_j} r_{i_j j} = \min_{1 \le i \le m} \{c_{ij} + \lambda_i r_{ij}\} ,$$

and an index t_j such as

$$c_{t_j j} + \lambda_{t_j} r_{t_j j} = \min_{\substack{1 \le i \le m \\ i \ne i_j}} \{c_{ij} + \lambda_i r_{ij}\} .$$

Then a solution $x(\lambda)$ of (2.3) is specified by

$$x_{i_j j}(\lambda) = 1, \quad x_{ij}(\lambda) = 0 \qquad \text{for all } i \ne i_j , \ 1 \le j \le m .$$

The vector $g(\lambda) \in \mathbb{R}^m$, defined as $g_i(\lambda) = \sum_{j=1}^{n} r_{ij} x_{ij}(\lambda) - b_i$, $1 \le i \le m$, is a subgradient of D at λ. Let

$$g_s(\lambda) = \max_{1 \le i \le m} \{g_i(\lambda)\} .$$

If $g_s(\lambda) \le 0$, then $x(\lambda)$ is primal-feasible, and, since it has been found in solving the dual, one hopes that it is near optimal. Thus, the stopping criterion for the procedure is $g_s(\lambda) \le 0$. If $g_s(\lambda) > 0$, then in Step 3 of the algorithm, the value of λ_s is increased at least to a point that one pair $x_{i_j j}, x_{t_j j}$ is "exchanged". Let the value of λ_s be increased by ρ_s,

i) if $i_j \ne s$, then

$$c_{sj} + (\lambda_s + \rho_s) r_{sj} \ge c_{ij} + \lambda_s r_{sj} \ge c_{i_j j} + \lambda_{i_j} r_{i_j j}$$

and x_{ij} remains unchanged for all i, $1 \le i \le m$;

ii) if $i_j = s$, then x_{ij} remains unchanged if

$$c_{sj} + (\lambda_s + \rho_s) r_{sj} \le c_{t_j j} + \lambda_{t_j} r_{t_j j} .$$

Thus, $\rho_s \leq \min_{j \in J_s} \left\{ \dfrac{(c_{t_j j} + \lambda_{t_j} r_{t_j j}) - (c_{sj} + \lambda_s r_{sj})}{r_{sj}} \right. = \Delta\lambda_s$

implies x does not change of state (where $J_s = \{j : i_j = s\}$).
We then take a stepsize equal to $\Delta\lambda_s + \varepsilon$. So, $x_{i_j j}$ becomes 0 and $x_{t_j j}$ becomes 1 for at least one $j \in J_s$.

To guarantee convergence, Klastorin introduces a $(m+1)^{st}$ index i (corresponding to a dummy building) such as for all j, $1 \leq j \leq n$,

$$c_{m+1\,j} = C \quad , \quad \text{a large scalar,}$$

$$r_{m+1\,j} = 1 \quad ,$$

$$b_{m+1} = n \quad .$$

This expanded problem is always feasible, but the original problem is feasible if and only if $x_{m+1\,j} = 0$, $1 \leq j \leq n$, at the last iteration. If this condition is not satisfied, Klastorin gives a heuristic search to recover a feasible solution in some cases.

Other variants of the algorithm were tested. They differ by the choice of the dual variable to modify in part a) of Step 3:

i) Klastorin's variant (K-variant) is summarized above.

ii) V1 - variant: Let

$$|g_s(\lambda)| = \max_{1 \leq i \leq m} \{|g_i(\lambda)|\} .$$

If $g_s(\lambda) < 0$, $g_i(\lambda) > 0$ for at least one i, $1 \leq j \leq m$, and $\lambda_s > 0$, then the value of λ_s is decreased, and an appropriate stepsize is determined. Otherwise K-variant is applied.

Notice the analogy of this variant with the "away step" variant [10] of the Frank-Wolfe technique in differentiable optimization.

iii) V2-variant: The K-variant is applied, but at each k^{th} iteration, let

$$g_s(\lambda) = \min_{1 \le i \le m} \{g_i(\lambda)\} ,$$

and if $g_s(\lambda) < 0$, and $\lambda_s > 0$, then the value of λ_s is decreased. Otherwise, K-variant is applied. The variant was tested with several values of k.

Table 1 summarizes the numerical experimentation. The results for the V2-variant with $k = 2$ are given in this table. For the class of problems generated, the three variants were competitive.

No comparison is made with the general subgradient algorithm because no estimate of the optimal value of $D(\lambda)$ is available during the execution of the algorithm. Since Polyak [8] insists on the superiority of the step-size determined using this estimate, it would not be fair to use any stepsize illustrating the superiority of our approach.

Ross and Soland [9] give a branch-and-bound algorithm to solve (2.1). This is an exact algorithm. Ross and Soland and Greedy require similar C.-P. execution time for problems of smaller size, but for problem 40 x 60, Ross and Soland requires 8 seconds to determine an optimal solution.

Problem size	C.P. time (sec.)			Number of iterations			Percentage of gap		
	K	V1	V2	K	V1	V2	K	V1	V2
10 x 50	0.56	0.58	0.85	27	28	42	8.8	8.6	13.0
20 x 30	0.36	0.37	0.60	16	16	26	6.3	7.8	12.0
20 x 40	0.35	0.46	0.48	11	15	30	5.1	5.8	5.8
20 x 50	0.61	1.31	1.32	16	38	34	4.9	9.9	8.7
30 x 50	0.51	1.21	1.10	8.20	24	21	2.1	5.3	3.6
20 x 60	1.04	0.98	2.21	24	23	54	4.6	6.0	10.4
40 x 60	2.10	2.32	1.65	26	30	20	3.6	4.1	2.9

TABLE 1

3. Selection - Assignment Problem

The problem is specified as:

$$\min \quad z = \sum_{i=1}^{m} \sum_{j=1}^{m} c_{ij}\, x_{ij}$$

$$\sum_{k=1}^{m} x_{ik} = y_i \,, \qquad 1 \le i \le m \,,$$

(3.1)

$$\sum_{k=1}^{m} x_{ki} = y_i \,, \qquad 1 \le i \le m \,,$$

$$\sum_{k \in Q_p} y_k = 1, \qquad 1 \le p \le n \,,$$

$$x_{ij} \text{ and } y_i = 0 \text{ or } 1, \qquad 1 \le i \le m;\ 1 \le j \le m,$$

where $Q_1, \ldots, Q_n$ is a partition of the set $\{1, \ldots, m\}$.

For a fixed feasible value of the vector y, the problem in x variables reduces to a $n \times n$ assignment problem. A problem with this structure arose in TRANSCOL [1], a school bus transportation study. Given a set of schools with several bus runs specified for each of them, the problem is to specify, within some time interval, the departure time for each run with the objective of minimizing the total number of buses used.

A Lagrangean dual of (3.1) is

(3.2) $\qquad \text{Max } D(\lambda)$

where

$$(3.3) \quad D(\lambda) = \min_{x,y \in X} \Big\{ \sum_{i=1}^{m} \sum_{j=1}^{m} c_{ij}\, x_{ij} + \sum_{i=1}^{m} \Big[\lambda_i \Big(\sum_{k=1}^{m} x_{ik} - y_i \Big) + \lambda_{m+i} \Big(\sum_{k=1} x_{ki} - y_i \Big) \Big] \Big\}$$

and $X=\{x,y: \sum_{k\in Q_p} y_k=1,\ 1\le p\le n,\ y_i=0 \text{ or } 1,\ x_{ij}=0 \text{ or } 1,\ 1\le i\le m,\ 1\le j\le m\}$.

As before, for fixed values of λ, Step 1 is easily performed if (3.3) is rewritten as

$$(3.4)\quad D(\lambda) = \min_{x,y\in X} \{\sum_{i=1}^{m}\sum_{j=1}^{m} (c_{ij}+\lambda_i+\lambda_{m+j})\, x_{ij} - \sum_{i=1}^{m} (\lambda_i+\lambda_{m+i})\, y_i\}\ .$$

A solution of (3.4) is then:

$$x_{ij}(\lambda) = \begin{cases} 1 & \text{if } c_{ij}+\lambda_i+\lambda_{m+j} \le 0 \\ 0 & \text{otherwise} \end{cases}$$

$$y_{i_p}(\lambda) = 1, \qquad y_i(\lambda) = 0 \quad i \ne i_p\ ,\ 1\le i\le m,\ 1\le p\le n\ ,$$

where the index i_p satisfies $\lambda_{i_p}+\lambda_{m+i_p} = \max_{i\in Q_p} \{\lambda_i+\lambda_{m+i}\}$.

The vector $g(\lambda)\in \mathbb{R}^{2m}$ defined by

$$g_i(\lambda) = \sum_{k=1}^{m} x_{ik}(\lambda) - y_i(\lambda)$$

$$g_{m+i}(\lambda) = \sum_{k=1}^{m} x_{ki}(\lambda) - y_i(\lambda)$$

is a subgradient of D at λ.

As usual, only one component of λ is modified in step 3.

Let $|g_s(\lambda)| = \max_{1\le i\le 2m} |g_i(\lambda)|$. Then the value of λ_s will be modified. To determine the stepsize, two different cases must be analyzed:

i) $g_s(\lambda)>0$. [The value of λ_s is increased.]

a) If $y_s(\lambda)=1$, then $\lambda_s+\lambda_{m+s} = \max_{i\in Q_{p_s}} \{\lambda_i+\lambda_{m+i}\}$

so increase λ_s until $\sum_{k=1}^{m} x_{sk}(\lambda) \le 1$ (or $\sum_{k=1}^{m} x_{ks}(\lambda) \le 1$ if $s>m$).

b) Otherwise, if $y_s(\lambda)=0$, increase λ_s until either $y_s(\lambda)$ increases to 1 or $\sum_{k=1}^{m} x_{sk}(\lambda) \le 1$ (or $\sum_{k=1}^{m} x_{ks}(\lambda) \le 1$, if $s>m$).

ii) $g_s(\lambda)<0$. [The value of λ_s is decreased.]

It is easy to see that $g_s(\lambda)<0$ implies that $y_s(\lambda)=1$ and $x_{sk}(\lambda)=0$, $1 \le k \le m$ (or $x_{ks}(\lambda)=0$ if $s>m$).

Thus, decrease λ_s until either

$y_s(\lambda) = 0$ or

$x_{sk}(\lambda) = 1$ ($x_{ks}(\lambda)=1$ if $s>m$) for some k.

One can see that an interval $[\underline{\lambda}_s, \overline{\lambda}_s]$ exists such as the value of $g_s(\lambda)$ remains equal to 0 for all $\lambda_s \in [\underline{\lambda}_s, \overline{\lambda}_s]$. Numerical experience indicates that it is more efficient to choose $\lambda_s := \frac{1}{2}(\underline{\lambda}_s + \overline{\lambda}_s)$.

To specify a stopping criterion, recall that the aim of the procedure is to solve the primal (3.1) and that $x(\lambda)$ and $y(\lambda)$ are primal-feasible if and only if $g(\lambda)=0$. For some problems, this might be difficult to achieve because of the duality gap. To obviate this difficulty, the corresponding assignment problem is solved every time a component of $y(\lambda)$ is modified. We now can use a function of (YP - YD)/YD as the stopping criterion, where YP and YD refer to the best primal and dual values of the objectives yet encountered.

Let us mention further improvements of the technique.

A good initial solution to (3.2) is:

$$\lambda_i = -\min_{1 \le j \le m} \{c_{ij}/2\} \quad , \qquad 1 \le i \le m \ ,$$

$$\lambda_{m+j} = -\min_{1 \le i \le m} \{c_{ij}/2\} \quad , \qquad 1 \le j \le m \ .$$

Thus, a variable x_{ij} will be set to 1 if and only if its coefficient is minimum in both its row and its column.

Duality gap seems unavoidable in many problems. Some modifications of the original data may eliminate some causes of the gap without modifying the solution. For instance, if i and j, $i \neq j$, belong to the same set Q_p, then the constraints of (3.1) implies that $x_{ij} = 0$. Thus, the solution is not modified if we replace c_{ij} by $+\infty$ whenever i and j, $i \neq j$, belong to the same set Q_p, but this modification improves the efficiency of the algorithm.

Table 2 compares the Greedy algorithm to a general subgradient optimization algorithm. The greedy approach is increasingly attractive as problem dimension grows up.

An interesting as well as surprising feature of both algorithms is that, for problems of a given size m, the C.P. time decreases when the number of classes n increases. But, on the other hand, for problems of size m having the same number of variables in each class, the number of feasible points increases with n at least until n reaches the value m/3. Indeed, if Q denotes the number of variables in each class, this follows from the fact that $Q = e$ (2.71827...) maximizes Q^n subject to the constraint $nQ = m$ and from the fact that the function $(\frac{m}{n})^n$ is quasi-concave in n. Hence our algorithm is faster for larger problems than for small ones.

Size	GREEDY			SUBGRADIENT		
	C.P. Time	Iter.	GAP(%)	C.P. time	Iter.	GAP(%)
10 x 3	0.27	44	12	0.10	8	6
10 x 5	0.27	28	10	0.14	6	7
10 x 2	0.12	23	5	0.07	6	6
20 x 3	0.91	85	13	0.79	23	9
20 x 5	0.38	29	9	0.40	13	8
20 x 7	0.19	13	7	0.42	10	7
30 x 5	1.24	66	8	1.43	33	8
30 x 7	1.84	80	12	2.49	46	11
30 x 9	0.89	33	9	1.81	27	8
50 x 9	3.88	83	9	2.71	23	8
50 x 11	2.56	49	10	1.74	13	8
50 x 15	1.49	23	9	1.13	6	9
100 x 15	15.24	100	9	9.48	24	9
100 x 20	8.34	51	10	2.33	4	8
100 x 25	1.23	5	8	1.26	1	7

Table 2

C.P. time is in seconds
Iter. is the number of iterations
GAP(%) is the percentage of gap

4. A min - max 0-1 Problem

The problem is

$$
\begin{aligned}
&\min \quad y_0 \\
&\text{s.t.} \quad \sum_{j=1}^{n} \sum_{i=1}^{I_j} b_{ij}^k \, x_{ij} \le y_0 \ , \qquad 1 \le k \le m \ , \\
&\qquad \sum_{i=1}^{I_j} x_{ij} = 1 \ , \qquad 1 \le j \le n \ , \\
&\qquad x_{ij} = 0 \text{ or } 1 \ , \qquad 1 \le i \le I_j \ , \ 1 \le j \le n \ .
\end{aligned} \tag{4.1}
$$

The problem is to minimize the maximum value of m linear forms under the constraint that exactly one variable in each of the n blocks is equal to 1. Problems with this structure arose in [1] and [3].

A Lagrangean dual of (4.1) can be specified:

$$\max_{\lambda \in \Lambda} \quad D(\lambda) \tag{4.2}$$

$$D(\lambda) = \min_{x \in X} \{ \sum_{k=1}^{m} \lambda_k \sum_{j=1}^{n} \sum_{i=1}^{I_j} b_{ij}^k \, x_{ij} \} \ , \tag{4.3}$$

and

$$X = \{x : \sum_{i=1}^{I_j} x_{ij} = 1, \ 1 \le j \le n, \ x_{ij} = 0 \text{ or } 1, \ 1 \le i \le I_j, \ 1 \le j \le n\},$$

$$\Lambda = \{\lambda : \sum_{k=1}^{m} \lambda_k = 1, \ \lambda_k \ge 0, \ 1 \le k \le m\} \ .$$

For a fixed value of λ, we can rewrite D as

$$D(\lambda) = \min_{x \in X} \{ \sum_{j=1}^{n} \sum_{i=1}^{I_j} (\sum_{k=1}^{m} \lambda_k \, b_{ij}^k) \, x_{ij} \} \tag{4.4}$$

Hence, for all j, $1 \le j \le n$, determine an index i_j such as

$$\sum_{k=1}^{m} \lambda_k b^k_{i_j j} = \min_{1 \le i \le I_j} \{ \sum_{k=1}^{m} \lambda_k b^k_{ij} \}$$

and an index t_j such as

$$\sum_{k=1}^{m} \lambda_k b^k_{t_j j} = \min_{\substack{1 \le i \le I_j \\ i \ne i_j}} \{ \sum_{k=1}^{m} \lambda_k b^k_{ij} \}$$

Then, a solution of (4.3) is specified by

$$x_{i_j j}(\lambda) = 1, \quad x_{ij}(\lambda) = 0, \quad i \ne i_j, \quad 1 \le i \le n .$$

The vector $g(\lambda) \in \mathbb{R}^m$ defined by

$$g_k(\lambda) = \sum_{j=1}^{n} \sum_{i=1}^{I_j} b^k_{ij} \, x_{ij}(\lambda)$$

is a subgradient of D at λ. Let $g_s(\lambda) = \max_{1 \le k \le m} \{g_k(\lambda)\}$.

In step 3, the value of λ_s is increased in such a way that $x_{i_j j}$ and $x_{t_j j}$ are "interchanged" in one and only one block. Let the value of λ_s be increased by ρ_s. For each j, $1 \le j \le n$, we expect that the variables x_{ij} will remain unchanged for all i, $1 \le i \le I_j$ if

$$\sum_{k=1}^{m} \lambda_k b^k_{i_j j} + \rho_s b^s_{i_j j} \le \sum_{k=1}^{m} \lambda_k b^k_{t_j j} + \rho_s b^s_{t_j j} .$$

Thus, ρ_s should verify

$$\rho_s \le \min_{j \in J_s} \left\{ \frac{\sum_{k=1}^{m} \lambda_k (b^k_{t_j j} - b^k_{i_j j})}{b^s_{i_j j} - b^s_{t_j j}} = \Delta\lambda_s \right.$$

(where $J_s = \{j : b^s_{i_j j} > b^s_{t_j j}, \ 1 \le i \le n\}$) for x_{ij} to remain the same; the

stepsize will then be

$$\lambda_s = \frac{\lambda_s + \Delta\lambda_s + \varepsilon}{1 + \Delta\lambda_s + \varepsilon}$$

$$\lambda_k = \frac{\lambda_k}{1 + \Delta\lambda_s + \varepsilon} \quad , \quad k \neq s \quad .$$

Furthermore, $x(\lambda)$ is primal feasible at each iteration. The stopping criterion is function of (YP-YD)/YD, where YP and YD refer to the best primal and dual values of the objectives yet encountered.

Table 3 compares the results obtained with the following algorithm: Greedy, Classical subgradient and Ferland-Florian. The Ferland-Florian algorithm [2] is a primal technique to solve (4.1). This algorithm determines better solutions than Greedy, and is also faster. Unfortunately, the bounds on the errors are useless for the Ferland-Florian algorithm. Greedy is superior than classical subgradient.

Problem size			C.P.time (sec)			Number of iterations		Solutions values		Percentage of Gap		
m	$\sum_{ij}$	n	G	C	F-F	G	C	G	F-F	G	C	F-F
40	120	40	2.3	17.9	2.3	11	150	1785	1685	14	16	31
30	120	30	3.3	9.2	1.3	24	111	1327	1260	14	16	36
20	120	20	2.1	3.3	1.0	23	66	866	832	13	15	35
40	90	40	1.9	14.2	1.6	11	136	1773	1688	14	15	33
30	90	30	1.3	6.1	0.9	11	92	1331	1267	14	15	37
20	60	20	0.5	2.4	0.3	8	70	889	845	14	15	30

Table 3

5. Conclusion

We presented a general method to solve integer programs, then we adapted it to 0-1 linear programs, and finally, we filled the details to use the method for three specific problems (as examples). The computer coding is straightforward. A strong point is that the algorithm gives a bound on the error of the solution. This bound is conservative.

Numerical results were encouraging and comparisons of Greedy with other methods show that the method is a viable alternative.

The extension of the Greedy method to general integer programs is not straightforward. It is not clear how to treat unbounded variables; moreover, even for bounded variables, step 1 must be worked out precisely to avoid "extreme" solutions where a variable is set to its lower (higher) bound if its coefficient is positive (negative).

REFERENCES

[1] DESROSIERS, J., Ferland J.-A., ROUSSEAU, J.-M. Horaires scolaires et circuits d'autobus scolaires. Document de travail #88, département d'informatique et de recherche opérationnelle, Université de Montréal,(mars 1977).

[2] FERLAND, J.-A., Florian, M. A Sub-Optimal Algorithm to Solve a Large Scale 0-1 Programming Problem. Proceedings of the XI International Symposium on Mathematical Programming,(1979).

[3] FLORIAN, M., BUSHELL, G., FERLAND, J., GUERIN, G., NASTANSKY, L. The Engine Scheduling Problem in a Railway Network. INFOR 14 pp. 121-138 (1976).

[4] GOFFIN, J.-L. On convergence Rates of Subgradient Optimisation Methods. Mathematical Programming 13, 321-347 (1977).

[5] HELD, M., WOLFE, P., CROWDER, H.P. Validation of Subgradient Optimization. Mathematical Programming 6, 62-68 (1978).

[6] KLASTORIN, T.D. An Effective Subgradient Algorithm for the Generalized Assignment Problem. University of Washington,(January 1977).

[7] POLYAK, B.T. A General Methof of Solving Extremes Problems. Soviet Math. Doklody 8, 593-597 (1967).

[8] POLYAK, B.T. Minimization of Unsmooth Functionals. U.S.S.R. Comp. Maths and Math. Physics 9, 14-29 (1969).

[9] ROSS, G.T., SOLAND, R.M. A Branch and Bound Algorithm for the Generalized Assignment Problem. Mathematical Programming 8, 91-103 (1975).

[10] WOLFE, P. Convergence Theory in Nonlinear Programming. In Integer and Nonlinear Programming, edited by J.A. Abadie, North Holland pp. 1-36 (1970).